Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey.

Laboratorio 20: Consultas en SQL
Construcción de software y toma de decisiones
TC2005B

Lilian Rodríguez Uribe | A01711949

Ximena Pérez Escalante | A

Ricardo Calzada Hernández | A01707647

1. **Consulta de un tabla completa**

   Algebra relacional.
   materiales

   SQL
   select * from materiales

	clave	descripcion	precio	impuesto
▶	1000	Varilla 3/16	100	10
	1010	Varilla 4/32	115	11.5
	1020	Varilla 3/17	130	13

5 09:31:49 select * from materiales LIMIT 0, 1000 45 row(s) returned

2. **Selección**

   Algebra relacional.
   SL{clave=1000}(materiales)

   SQL
   select * from materiales
   where clave=1000

	clave	descripcion	precio	impuesto
▶	1000	Varilla 3/16	100	10

7 09:34:49 select * from materiales where clave=1000 LIMIT 0, 1000 1 row(s) returned

3. **Proyección**

Algebra relacional.
PR{clave,rfc,fecha} (entregan)

SQL
select clave,rfc,fecha from entregan

	clave	rfc	fecha
▶	1000	AAAA800101	2001-12-13
	1200	EEEE800101	2003-03-15
	1400	AAAA800101	1999-04-07

✓	8	09:36:09	select clave,rfc,fecha from entregan LIMIT 0, 1000	87 row(s) returned

4. **Reunión Natural**

Algebra relacional.
entregan JN materiales

SQL
select * from materiales,entregan
where materiales.clave = entregan.clave

	clave	descripcion	precio	impuesto	clave	rfc	numero	fecha	cantidad
▶	1000	Varilla 3/16	100	10	1000	AAAA800101	5000	2001-12-13	165
	1000	Varilla 3/16	100	10	1000	AAAA800101	5019	1999-07-13	254
	1010	Varilla 4/32	115	11.5	1010	BBBB800101	5001	1998-07-28	528

✓	9	09:40:09	select * from materiales,entregan where materiales.clave = entrega...	87 row(s) returned

Si algún material no ha se ha entregado ¿Aparecería en el resultado de esta consulta?
No, únicamente se muestran aquellos materiales entregados (por ello deben coincidir las claves de las tablas materiales y entregan).

5. **Reunión con criterio específico**

Algebra relacional.
entregan JN{entregan.numero <= proyectos.numero} proyectos

SQL
select * from entregan,proyectos
where entregan.numero < = proyectos.numero

	clave	rfc	numero	fecha	cantidad	numero	denominacion
▶	1000	AAAA800101	5000	2001-12-13	165	5000	Vamos Mexico
	1200	EEEE800101	5000	2003-03-15	177	5000	Vamos Mexico
	1400	AAAA800101	5000	1999-04-07	382	5000	Vamos Mexico

✓	10	09:42:58	select * from entregan,proyectos where entregan.numero <= proye...	836 row(s) returned

6. **Unión (se ilustra junto con selección)**

Algebra relacional.

SL{clave=1450}(entregan) UN SL{clave=1300}(entregan)

SQL
(select * from entregan where clave=1450)
union
(select * from entregan where clave=1300)

	clave	rfc	numero	fecha	cantidad
▶	1300	GGGG800101	5005	2004-02-28	521
	1300	GGGG800101	5010	2001-02-10	119

11 09:44:32 (select * from entregan where clave=1450) union (select * from entr... 2 row(s) returned

¿Cuál sería una consulta que obtuviera el mismo resultado sin usar el operador Unión? Compruébalo.
select * from entregan where clave = 1450 or clave = 1300

La consulta sería la siguiente

	clave	rfc	numero	fecha	cantidad
▶	1300	GGGG800101	5005	2004-02-28	521
	1300	GGGG800101	5010	2001-02-10	119

12 09:46:03 select * from entregan where clave = 1450 or clave = 1300 LIMIT ... 2 row(s) returned

7. **Intersección (se ilustra junto con selección y proyección)**

Algebra relacional.
PR{clave}(SL{numero=5001}(entregan)) IN PR{clave}(SL{numero=5018}(entregan))

SQL
Nota: Debido a que en SQL server no tiene definida alguna palabra reservada que nos permita hacer esto de una manera entendible, veremos esta sección en el siguiente laboratorio con el uso de Subconsultas. Un ejemplo de un DBMS que si tiene la implementación de una palabra reservada para esta función es Oracle, en él si se podría generar la consulta con una sintaxis como la siguiente:

(select clave from entregan where numero=5001)
intersect
(select clave from entregan where numero=5018)

	clave
▶	1010

13 09:47:11 (select clave from entregan where numero=5001) intersect (select ... 1 row(s) returned

8. **Diferencia (se ilustra con selección)**

Algebra relacional.
entregan - SL{clave=1000}(entregan)

SQL
(select * from entregan)
minus
(select * from entregan where clave=1000)

Nuevamente, "minus" es una palabra reservada que no está definida en SQL Server, define una consulta que regrese el mismo resultado.

Utilizando:
SELECT * FROM entregan
WHERE clave NOT IN (
SELECT clave
FROM entregan WHERE
clave = 1000);

La consulta sería la siguiente:

	clave	rfc	numero	fecha	cantidad
▶	1010	BBBB800101	5001	1998-07-28	528
	1010	BBBB800101	5018	1997-02-09	523
	1020	CCCC800101	5002	2003-12-16	582

15 09:56:45 SELECT * FROM entregan WHERE clave NOT IN (SELECT cla... 85 row(s) returned

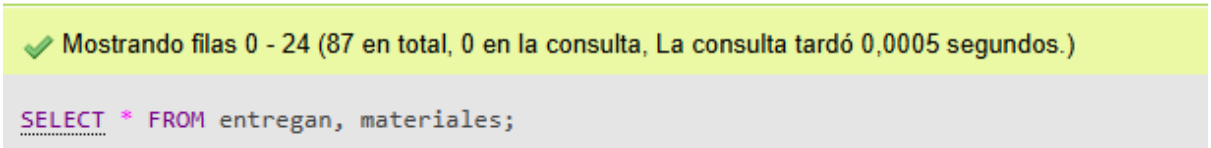
9. **Producto cartesiano**

Algebra relacional.
entregan X materiales

SQL
select * from entregan,materiales

¿Cómo está definido el número de tuplas de este resultado en términos del número de tuplas de entregan y de materiales?

clave	rfc	numero	fecha	cantidad	clave	descripcion	precio	impuesto
1000	AAAA800101	5000	2001-12-13	165	1000	Varilla 3/16	100	10
1000	AAAA800101	5000	2001-12-13	165	1010	Varilla 4/32	115	11.5
1000	AAAA800101	5000	2001-12-13	165	1020	Varilla 3/17	130	13
1000	AAAA800101	5000	2001-12-13	165	1030	Varilla 4/33	145	14.5



El número de tuplas de esta consulta está definido en base a la suma de tuplas de las dos tablas seleccionadas en el comando las cuales son entregan y materiales

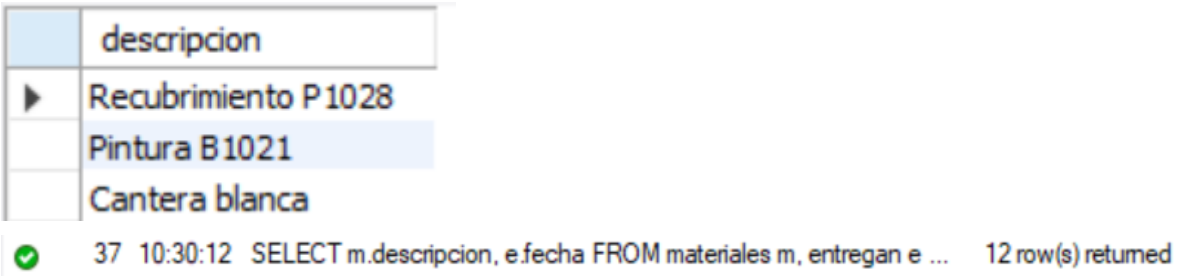
10. Construcción de consultas a partir de una especificación

Plantea ahora una consulta para obtener las descripciones de los materiales entregados en el año 2000.

Recuerda que la fecha puede indicarse como '01-JAN-2000' o '01/01/00'.

Importante: Recuerda que cuando vayas a trabajar con fechas, antes de que realices tus consultas debes ejecutar la instrucción "set dateformat dmy". Basta con que la ejecutes una sola vez para que el manejador sepa que vas a trabajar con ese formato de fechas.

La consulta es la siguiente:
SELECT m.descripcion, e.fecha
FROM materiales m, entregan e
WHERE m.clave = e.clave
AND e.fecha BETWEEN '2000-01-01' AND '2000-12-31'

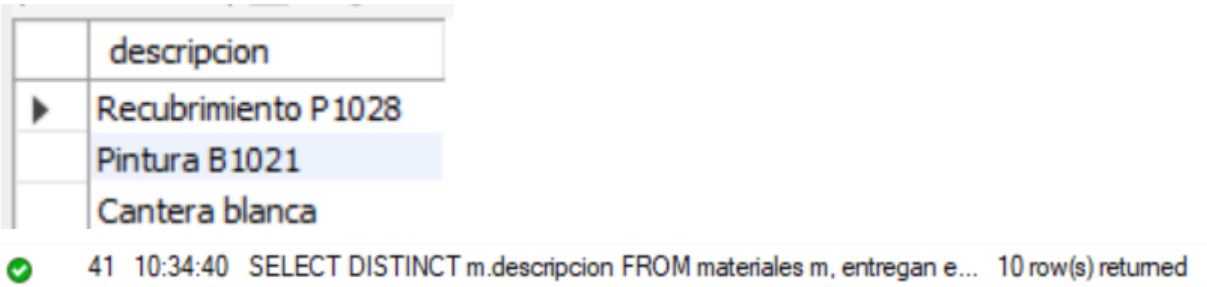


¿Por qué aparecen varias veces algunas descripciones de material?
Porque cada entrega en entregan genera una fila en el resultado, incluso si la descripción en materiales es la misma.

11. Uso del calificador distinct

En el resultado anterior, observamos que una misma descripción de material aparece varias veces.

Agrega la palabra distinct inmediatamente después de la palabra select a la consulta que planteaste antes.



¿Qué resultado obtienes en esta ocasión?
Se obtiene una única vez cada material.

12. Ordenamientos.

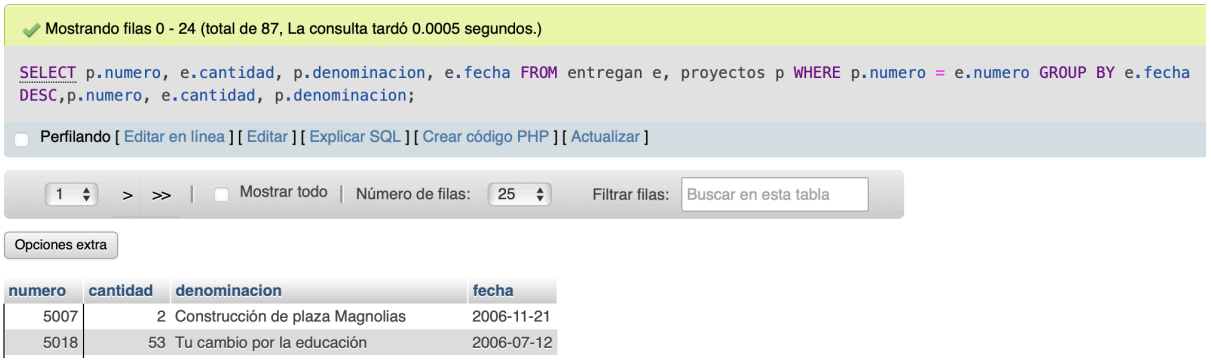
Si al final de una sentencia select se agrega la cláusula

order by campo [desc] [,campo [desc] ...]

donde las partes encerradas entre corchetes son opcionales (los corchetes no forman parte de la sintaxis), los puntos suspensivos indican que pueden incluirse varios campos y la palabra desc se refiere a descendente. Esta cláusula permite presentar los resultados en un orden específico.

Obtén los números y denominaciones de los proyectos con las fechas y cantidades de sus entregas, ordenadas por número de proyecto, presentando las fechas de la más reciente a la más antigua.

La consulta es la siguiente:
SELECT p.numero, e.cantidad, p.denominacion, e.fecha
FROM entregan e, proyectos p
WHERE p.numero = e.numero
GROUP BY e.fecha DESC,p.numero, e.cantidad, p.denominacion;



13. Uso de expresiones.

En álgebra relacional los argumentos de una proyección deben ser columnas. Sin embargo en una sentencia SELECT es posible incluir expresiones aritméticas o funciones que usen

como argumentos de las columnas de las tablas involucradas o bien constantes. Los operadores son:

+ Suma
- Resta
* Producto
/ División

Las columnas con expresiones pueden renombrarse escribiendo después de la expresión un alias que puede ser un nombre arbitrario; si el alias contiene caracteres que no sean números o letras (espacios, puntos etc.) debe encerrarse entre comillas dobles (" nuevo nombre"). Para SQL Server también pueden utilizarse comillas simples.

14. Operadores de cadena

El operador LIKE se aplica a datos de tipo cadena y se usa para buscar registros, es capaz de hallar coincidencias dentro de una cadena bajo un patrón dado.

También contamos con el operador comodín (%), que coincide con cualquier cadena que tenga cero o más caracteres. Este puede usarse tanto de prefijo como sufijo.

SELECT * FROM productos where Descripcion LIKE 'Si%'

¿Qué resultado obtienes?

	clave	descripcion	precio	impuesto
▶	1120	Sillar rosa	100	10
	1130	Sillar gris	110	11

45 10:40:29 SELECT * FROM materiales where Descripcion LIKE '... 2 row(s) returned

Explica que hace el símbolo '%'.
Indica que después del patrón indicado, puede o no haber una cadena
¿Qué sucede si la consulta fuera : LIKE 'Si' ?
Buscaría únicamente registros en los que la descripción sea tal cual la cadena “Si”.
¿Qué resultado obtienes?
Ninguno, ya que no hay ningún material con esa descripción.
Explica a qué se debe este comportamiento.
Usar el operador LIKE busca el patrón en la columna indicada, si no agregamos operadores como %, LIKE buscará aquellos registros en los que los datos coincidan exactamente con el patrón.

Otro operador de cadenas es el de concatenación, (+, +=) este operador concatena dos o más cadenas de caracteres.
Su sintaxis es : Expresión + Expresión.
Un ejemplo de su uso, puede ser: Un ejemplo de su uso, puede ser:
SELECT (Apellido + ', ' + Nombre) as Nombre FROM Personas;

```
DECLARE @foo varchar(40);
DECLARE @bar varchar(40);
SET @foo = '¿Que resultado';
SET @bar = ' ¿¿¿??? '
SET @foo += ' obtienes?';
PRINT @foo + @bar;
```

¿Qué resultado obtienes de ejecutar el siguiente código?
‘¿Qué resultado obtienes?¿¿¿???’

	CONCAT(@foo, @bar)
▶	¿Que resultado obtienes? ¿¿¿???

¿Para qué sirve DECLARE?
Para definir variables.
¿Cuál es la función de @foo?
Es una variable que actúa como un contenedor para almacenar una cadena de texto.
¿Que realiza el operador SET?
Asigna un valor a una variable.
Sin embargo, tenemos otros operadores como [ ] , [^] y _.

[ ] - Busca coincidencia dentro de un intervalo o conjunto dado. Estos caracteres se pueden utilizar para buscar coincidencias de patrones como sucede con LIKE.

[^] - En contra parte, este operador coincide con cualquier caracter que no se encuentre dentro del intervalo o del conjunto especificado.

_ - El operador _ o guion bajo, se utiliza para coincidir con un caracter de una comparación de cadenas.

Ahora explica el comportamiento, función y resultado de cada una de las siguientes consultas:

SELECT RFC FROM Entregan WHERE RFC LIKE '[A-D]%';
SELECT RFC FROM Entregan WHERE RFC LIKE '[^A]%';
Si se mantiene como la consulta anterior, no se obtiene ningún registro, sin embargo, si se cambia a:
SELECT RFC FROM Entregan WHERE RFC LIKE 'A%'; se regresan los RFC que comiencen con A.
SELECT Numero FROM Entregan WHERE Numero LIKE '___6';
Se muestran los RFC que tengan tres caracteres (no importa cuáles) antes de un 6.

15. Operadores compuestos.

Los operadores compuestos ejecutan una operación y establecen un valor.
+ = (Suma igual)
- = (Restar igual)
* = (Multiplicar igual)
/ = (Dividir igual)
% = (Módulo igual)

16. Operadores Lógicos.

Los operadores lógicos comprueban la verdad de una condición, al igual que los operadores de comparación, devuelven un tipo de dato booleano (True, false o unknown).

ALL Es un operador que compara un valor numérico con un conjunto de valores representados por un subquery. La condición es verdadera cuando todo el conjunto cumple la condición.

ANY o SOME Es un operador que compara un valor numérico con un conjunto de valores. La condición es verdadera cuando al menos un dato del conjunto cumple la condición.

La sintaxis para ambos es: valor_numerico {operador de comparación} subquery

BETWEEN Es un operador para especificar intervalos. Una aplicación muy común de dicho operador son intervalos de fechas.

SELECT Clave,RFC,Numero,Fecha,Cantidad
FROM Entregan
WHERE Numero Between 5000 and 5010;

¿Cómo filtrarías rangos de fechas?
Utilizando between para establecer el rango, por ejemplo:

```
SELECT Clave,RFC,Numero,Fecha,Cantidad
FROM Entregan
WHERE fecha Between 2001-01-01 and 2004-01-01;
```

EXISTS Se utiliza para especificar dentro de una subconsulta la existencia de ciertas filas.

```
SELECT RFC,Cantidad, Fecha,Numero
FROM [Entregan]
WHERE [Numero] Between 5000 and 5010 AND
Exists ( SELECT [RFC]
FROM [Proveedores]
WHERE RazonSocial LIKE 'La%' and [Entregan].[RFC] = [Proveedores].[RFC] )
```

¿Qué hace la consulta?
Filtra las entregas con número entre 5000 y 5010, y muestra aquellas en las que la razón social del proveedor empieza con 'La'.
¿Qué función tiene el paréntesis ( ) después de EXISTS?
Delimita la subconsulta.

IN Especifica si un valor dado tiene coincidencias con algún valor de una subconsulta. NOTA: Se utiliza dentro del WHERE pero debe contener un parametro. Ejemplo: Where proyecto.id IN Lista_de_Proyectos_Subquery

Tomando de base la consulta anterior del EXISTS, realiza el query que devuelva el mismo resultado, pero usando el operador IN

```
SELECT rfc, cantidad, fecha, numero
FROM entregan
WHERE numero BETWEEN 5000 AND 5010
AND rfc IN (SELECT rfc
            FROM proveedores
            WHERE razonsocial LIKE 'La%')
```

NOT Simplemente niega la entrada de un valor booleano.

Tomando de base la consulta anterior del EXISTS, realiza el query que devuelva el mismo resultado, pero usando el operador NOT IN Realiza un ejemplo donde apliques algún operador : ALL, SOME o ANY.

```
SELECT rfc, cantidad, fecha, numero
FROM entregan
WHERE numero BETWEEN 5000 AND 5010
AND rfc NOT IN (SELECT rfc
          FROM proveedores
          WHERE razonsocial NOT LIKE 'La%')
```

El Operador TOP, es un operador que recorre la entrada, un query, y sólo devuelve el primer número o porcentaje especifico de filas basado en un criterio de ordenación si es posible.

¿Qué hace la siguiente sentencia? Explica por qué.

SELECT TOP 2 * FROM Proyectos

Devuelve las primeras 2 filas de la tabla Proyectos pero sin un orden definido ya que para esto se requiere un ORDER BY.

¿Qué sucede con la siguiente consulta? Explica por qué.

SELECT TOP Numero FROM Proyectos

Devuelve un error de sintaxis , ya que TOP no acepta una columna como argumento, requiere un número entero o una variable.

17. Modificando la estructura de un tabla existente.

Agrega a la tabla materiales la columna PorcentajeImpuesto con la instrucción:
ALTER TABLE materiales ADD PorcentajeImpuesto NUMERIC(6,2);
A fin de que los materiales tengan un impuesto, les asignaremos impuestos ficticios basados en sus claves con la instrucción:
UPDATE materiales SET PorcentajeImpuesto = 2*clave/1000;
esto es, a cada material se le asignará un impuesto igual al doble de su clave dividida entre diez.

Revisa la tabla de materiales para que compruebes lo que hicimos anteriormente.

18. Creación de vistas

La sentencia:

Create view nombrevista (nombrecolumna1 , nombrecolumna2 ,..., nombrecolumna3)
as select...

Permite definir una vista. Una vista puede pensarse como una consulta etiquetada con un nombre, ya que en realidad al referirnos a una vista el DBMS realmente ejecuta la consulta asociada a ella, pero por la cerradura del álgebra relacional, una consulta puede ser vista como una nueva relación o tabla, por lo que es perfectamente válido emitir la sentencia:

select * from nombrevista

¡Como si nombrevista fuera una tabla!

Comprueba lo anterior, creando vistas para cinco de las consultas que planteaste anteriormente en la práctica . Posteriormente revisa cada vista creada para comprobar que devuelve el mismo resultado.

SELECT * FROM VistaSeleccion

clave	descripcion	precio	impuesto	PorcentajeImpuesto
1000	Varilla 3/16	100	10	2.00

SELECT * FROM VistaProyeccion

clave	rfc	numero	fecha	cantidad
1000	AAAA800101	5000	2001-12-13	165
1000	AAAA800101	5019	1999-07-13	254
1010	BBBB800101	5001	1998-07-28	528

SELECT * FROM VistaReunionNatural

m_clave	descripcion	precio	impuesto	PorcentajeImpuesto	e_clave	rfc	numero	fecha	cantidad
1000	Varilla 3/16	100	10	2.00	1000	AAAA800101	5000	2001-12-13	165
1000	Varilla 3/16	100	10	2.00	1000	AAAA800101	5019	1999-07-13	254
1010	Varilla 4/32	115	11.5	2.02	1010	BBBB800101	5001	1998-07-28	528

SELECT * FROM VistaUnion

clave	rfc	numero	fecha	cantidad
1300	GGGG800101	5005	2004-02-28	521
1300	GGGG800101	5010	2001-02-10	119

SELECT * FROM VistaImporte

clave	cantidad	PorcentajeImpuesto	importe
1000	165	2.00	330.00
1000	254	2.00	508.00
1010	528	2.02	1066.56

La parte (nombrecolumna1,nombrecolumna2,.de la sentencia create view puede ser omitida si no hay ambigüedad en los nombres de las columnas de la sentencia select asociada.

Importante: Las vistas no pueden incluir la cláusula order by.

A continuación se te dan muchos enunciados de los cuales deberás generar su correspondiente consulta.

En el reporte incluye la sentencia, una muestra de la salida (dos o tres renglones) y el número de renglones que SQL Server reporta al final de la consulta.

Los materiales (clave y descripción) entregados al proyecto "México sin ti no estamos completos".

```
1 •  SELECT m.clave, m.descripcion
2    FROM materiales m, entregan e, proyectos p
3    WHERE m.clave = e.clave
4    AND e.numero = p.numero
5    AND p.denominacion = 'Mexico sin ti no estamos completos';
```

	clave	descripcion
▶	1030	Varilla 4/33
	1230	Cemento
	1430	Pintura B1022

26 21:06:25 SELECT m.clave, m.descripcion FROM materiales m, en... 3 row(s) returned

Los materiales (clave y descripción) que han sido proporcionados por el proveedor "Acme tools".

```
1 •  SELECT m.clave, m.descripcion
2    FROM materiales m, entregan e, proveedores p
3    WHERE m.clave = e.clave
4    AND e.rfc = p.rfc
5    AND p.razonsocial = 'Acme tools';
```

	clave	descripcion

29 21:14:54 SELECT m.clave, m.descripcion FROM materiales m, en... 0 row(s) returned

El RFC de los proveedores que durante el 2000 entregaron en promedio cuando menos 300 materiales.

```
SELECT rfc
FROM entregan
WHERE fecha BETWEEN '2000-01-01' AND '2000-12-31'
AND cantidad >= 300;
```

	rfc
▶	FFFF800101
	CCCC800101
	FFFF800101

32 21:29:12 SELECT rfc FROM entregan WHERE fecha BETWEEN... 5 row(s) returned

El Total entregado por cada material en el año 2000.

```
1  SELECT m.clave, m.descripcion, SUM(e.cantidad) AS total_entregado
2  FROM materiales m, entregan e
3  WHERE m.clave = e.clave
4  AND e.fecha BETWEEN '2000-01-01' AND '2000-12-31'
5  GROUP BY m.clave, m.descripcion;
```

clave	descripcion	total_entregado
1020	Varilla 3/17	8
1050	Varilla 4/34	623
1100	Block	466

39 21:38:11 SELECT m.clave, m.descripcion, SUM(e.cantidad) AS to... 11 row(s) returned

La Clave del material más vendido durante el 2001. (se recomienda usar una vista intermedia para su solución)

```
1  CREATE VIEW VistaTotales2001 (clave, total_entregado) AS
2  SELECT
3      e.clave,
4      SUM(e.cantidad) AS total_entregado
5  FROM entregan e
6  WHERE e.fecha BETWEEN '2001-01-01' AND '2001-12-31'
7  GROUP BY e.clave;

9  SELECT clave
10 FROM VistaTotales2001
11 WHERE total_entregado = (SELECT MAX(total_entregado) FROM VistaTotales2001);
```

clave
1260

41 21:49:21 SELECT clave FROM VistaTotales2001 WHERE total_entregado = (SELECT MAX(total_... 1 row(s) returned

Productos que contienen el patrón 'ub' en su nombre.

```
1  SELECT clave, descripcion
2  FROM materiales
3  WHERE descripcion LIKE '%ub%';
```

	clave	descripcion
▶	1180	Recubrimiento P1001
	1190	Recubrimiento P1010
	1200	Recubrimiento P1019

42 21:52:03 SELECT clave, descripcion FROM materiales WHERE d... 12 row(s) returned

Denominación, RFC y RazonSocial de los proveedores que se suministran materiales al proyecto Televisa en acción que no se encuentran apoyando al proyecto Educando en Coahuila (Solo usando vistas).

```
CREATE VIEW VistaTelevisa (rfc, razonsocial) AS
SELECT DISTINCT pr.rfc, pr.razonsocial
FROM proveedores pr, entregan e, proyectos p
WHERE pr.rfc = e.rfc
AND e.numero = p.numero
AND p.denominacion = 'Televisa en acción';
```

```
CREATE VIEW VistaEducando (rfc, razonsocial) AS
SELECT DISTINCT pr.rfc, pr.razonsocial
FROM proveedores pr, entregan e, proyectos p
WHERE pr.rfc = e.rfc
AND e.numero = p.numero
AND p.denominacion = 'Educando en Coahuila';
```

```
CREATE VIEW VistaProveedoresExclusivos (rfc, razonsocial, denominacion) AS
SELECT t.rfc, t.razonsocial, 'Televisa en acción' AS denominacion
FROM VistaTelevisa t
WHERE t.rfc NOT IN (SELECT rfc FROM VistaEducando);
```

```
SELECT * FROM VistaProveedoresExclusivos;
```

	rfc	razonsocial	denominacion
▶	DDDD800101	Cecoferre	Televisa en acción
	CCCC800101	La Ferre	Televisa en acción

48 22:05:26 SELECT * FROM VistaProveedoresExclusivos LIMIT 0, 1000 2 row(s) returned

Denominación, RFC y RazonSocial de los proveedores que se suministran materiales al proyecto Televisa en acción que no se encuentran apoyando al proyecto Educando en Coahuila (Sin usar vistas, utiliza not in, in o exists).

```sql
SELECT DISTINCT
    'Televisa en acción' AS denominacion,
    p.rfc,
    p.razonsocial
FROM proveedores p, entregan e, proyectos pr
WHERE p.rfc = e.rfc
AND e.numero = pr.numero
AND pr.denominacion = 'Televisa en acción'
AND p.rfc NOT IN (
    SELECT e2.rfc
    FROM entregan e2, proyectos pr2
    WHERE e2.numero = pr2.numero
    AND pr2.denominacion = 'Educando en Coahuila'
);
```

	denominacion	rfc	razonsocial
▶	Televisa en acción	DDDD800101	Cecoferre
	Televisa en acción	CCCC800101	La Ferre

50 22:17:32 SELECT DISTINCT 'Televisa en acción' AS denominacion, p.rfc, p.razonsocial F... 2 row(s) returned

Costo de los materiales y los Materiales que son entregados al proyecto Televisa en acción cuyos proveedores también suministran materiales al proyecto Educando en Coahuila.

```sql
SELECT DISTINCT m.clave, m.descripcion, m.precio AS costo
FROM materiales m, entregan e, proyectos p
WHERE m.clave = e.clave
AND e.numero = p.numero
AND p.denominacion = 'Televisa en acción'
AND e.rfc IN (
    SELECT e2.rfc
    FROM entregan e2, proyectos p2
    WHERE e2.numero = p2.numero
    AND p2.denominacion = 'Educando en Coahuila'
);
```

clave	descripcion	costo
1080	Ladrillos rojos	50
1280	Tepetate	34

51 22:20:45 SELECT DISTINCT m.clave, m.descripcion, m.precio AS costo FROM materiales m, entre... 2 row(s) returned

Reto: Usa solo el operador NOT IN en la consulta anterior (No es parte de la entrega).

Nombre del material, cantidad de veces entregados y total del costo de dichas entregas por material de todos los proyectos.

```sql
SELECT m.descripcion, COUNT(*) AS veces_entregado, SUM(e.cantidad * m.precio) AS total_costo
FROM materiales m, entregan e
WHERE m.clave = e.clave
GROUP BY m.descripcion;
```

descripcion	veces_entregado	total_costo
Varilla 3/16	2	41900
Varilla 4/32	2	120865
Varilla 3/17	2	76700

52 22:23:26 SELECT m.descripcion, COUNT(*) AS veces_entregado, SUM(e.cantidad * m.precio) AS t... 42 row(s) returned